

15

Lesson 15 Advanced Form Functions

Lesson Topics

- ▶ HTML Form Review
- ▶ Supplemental Field Attributes
- ▶ Embedding Tables in Forms
- ▶ Forms Tips & Tricks
- ▶ Posting Forms to CGI Programs
- ▶ Lesson 9 Summary

HTML Form Review

Lesson 14: *Introduction to HTML Forms* provided you with a solid knowledge of and practice creating forms. In this Lesson, you will extend your knowledge by using special attributes to modify the appearance or functionality of form fields. You will also mix different HTML elements, such as forms and tables, and forms and images.

Test Your Forms Knowledge

Before you move on to this Lesson, you need to test what you learned in the previous Lesson. Complete the quiz below.

1. What are the two types of form fields that do not rely on the `<INPUT>` tag?

2. Script the code for a list box that presents a user with the following options: High School Grad, Some College, College Grad, Post Graduate Work, Masters Degree, Ph.D., M.D.
3. True or False: when creating list boxes, adding `VALUE` attributes to the `<OPTION>` tags will cause the user's browser to misinterpret the code and ruin the entire form.
4. True or False: there is a limit of 12 items in a group of checkboxes.
5. By default, what is the limit to the amount of text a user can type into a text area field?
6. Script the code for a password field that accepts a credit card number.
7. Script the code for the form element that allows a user to upload a populated form.
8. What is the `<FORM>` attribute that determines the specific action taken when a user clicks the Submit button? _____
9. Radio buttons denote an exclusive selection logic; their cousins, _____, involve a non-exclusive logic in which the user can choose multiple or even all options.
10. Radio buttons and check boxes require the _____ attribute to the `<INPUT>` tag; other form fields do not.

Supplemental Field Attributes

You already know about the necessary (required) form tag attributes, such as NAME and—for checkboxes and radio buttons—VALUE.

However, there are also many useful and powerful supplemental attributes that can help you customize your form fields to best suit a particular application or user preferences.

Text & Password Fields

Text and password fields, defined by the TYPE=text and TYPE=password attributes to the <INPUT> tag, have two supplemental attributes, as shown in Table 15-1.

Attribute	Value Set
SIZE	Any positive integer. Translates into the width of the text field in characters. <u>Example:</u> <INPUT TYPE=password NAME=creditcard SIZE=16 >
MAXLENGTH	Any positive integer. Limits the number of characters that a user can input into the field, regardless of the SIZE which has been specified. <u>Example:</u> <INPUT TYPE=text NAME=zip MAXLENGTH=10 >

Table 15-1: Text and password field supplemental attributes

Characteristics

Remember the following characteristics of both SIZE and MAXLENGTH:

- The values of SIZE and MAXLENGTH do not have to equal each other.
- The value of MAXLENGTH does not have to be less than the value of SIZE.
- You can use either or both of these supplemental attributes when scripting a text field or a password field.
- Although not technically required, it is recommended that you always specify a MAXLENGTH value that is less than the SIZE value.

Exercise 15-1: Using the SIZE and MAXLENGTH Attributes

In this exercise, you will use the SIZE and MAXLENGTH attributes with both a text field and a password field.

1. Launch Notepad (or the ASCII text editor of your choice) and open SIZE_AND_MAXLENGTH.HTM from the HTML-3 folder on your Desktop.
2. Toggle over to your Web browser and open SIZE_AND_MAXLENGTH.HTM.
3. Toggle back to Notepad and add the following code that appears in bold:

```
First name: <INPUT TYPE=text NAME=first_name SIZE=10><BR>
Last name: <INPUT TYPE=text NAME=last_name SIZE=15><BR>
Address: <INPUT TYPE=text NAME=address SIZE=20><BR>
City: <INPUT TYPE=text NAME=city SIZE=15><BR>
State:      <SELECT NAME=state>
             <OPTION>New York
             <OPTION>Ohio
             <OPTION>Pennsylvania
             <OPTION>Maryland
             <OPTION>West Virginia
             <OPTION>Connecticut
             <OPTION>New Jersey
             </SELECT><BR>
Zip+4 Code: <INPUT TYPE=text NAME=zip MAXLENGTH=10 SIZE=11><P>

<HR WIDTH=50% ALIGN=left SIZE=6><P>

Social Security #: <INPUT TYPE=password NAME=ssn MAXLENGTH=11
SIZE=12><P>
```

4. Save the changes you have made to SIZE_AND_MAXLENGTH.HTM.
5. Toggle over to your Web browser. Note the change in the lengths of the text fields as you press <CTRL + R> to reload the Web page.



The lengths of the fields change according to the changes you made in Step 3.

6. Click once in the **Zip+4** field and type all nine characters of your zip code. Separate the first five characters and the +4 section by a hyphen. Note that you cannot type more text (the MAXLENGTH attribute prevents it).

7. Press <TAB> to advance to the **Social Security #** field and type your social security number, using hyphens where appropriate. Note: the contents of a password field will always appear as asterisks for security purposes.
8. Try to type an additional number in the **Social Security #** field. You cannot due to the value of the MAXLENGTH attribute. Depending on the configuration of your computer, you may hear an audible alert each time you attempt to type a character that exceeds the limit set by MAXLENGTH.



Note that the **SIZE** attribute values have been set to one character more than the value of the **MAXLENGTH** fields. This is solely for aesthetic purposes.

Remember: there is no technical relationship between the **SIZE** and **MAXLENGTH** values. They can be set to any positive integer independently.



The page in your browser should appear similar to Figure 15-1.

Adding SIZE & MAXLENGTH Attributes - Netscape

File Edit View Go Window Help

Back Forward Reload Home Search Netscape Print Security Stop

Bookmarks Location: file:///C:/WINNT/Profiles/Administrator/Desktop/Html-3/size_and_maxlength.htm

Using SIZE & MAXLENGTH Attributes

First name:

Last name:

Address:

City:

State:

Zip+4 Code:

Social Security #:

Document Done

Figure 15-1: Using **SIZE** and **MAXLENGTH** to restrict input in text and password fields

Checkboxes & Radio Buttons

Checkboxes and radio buttons have a unique supplemental attribute with which you can create more specialized forms for particular applications and topics.

This supplemental attribute is described in Table 15-2.

Attribute	Value Set
CHECKED	<p>None. By adding CHECKED to the <INPUT> tag, the checkbox or radio button will be preselected (with no user action). The user can still deselect or reselect the field.</p> <p><u>Example:</u></p> <pre><INPUT TYPE=radio NAME=age VALUE=teen CHECKED></pre>

Table 15-2: Checkbox and radio button CHECKED attribute



“At its heart, HTML is about structure. Author are given a set of tools to mark the purpose and intent of each piece of text within a Web page. It’s then up to the browser to decide how to display the content to the end user. By working with structure separately from appearance, HTML comes closer to achieving true independence from the proprietary demand of specific browsers.”

— Rick Darnell, *Internet Writer*, 1998

Exercise 15-2: Using the CHECKED Attribute

In this exercise, you will apply the CHECKED attribute to both a set of checkboxes and a set of radio buttons.

1. In your Web browser, open CHECKED.HTM from the HTML-3 folder.
2. Toggle over to your text editor and open CHECKED.HTM.
3. Add the following code that appears in bold:

```
<FONT FACE=verdana COLOR=green SIZE=3><B>Favorite Form of
Exercise</B></FONT><P>

<INPUT TYPE=checkbox NAME=sport VALUE=run CHECKED> Running<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=cycle> Bicycling<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=aerobics> Aerobics<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=swim> Swimming<BR>
<INPUT TYPE=checkbox NAME=sport VALUE=TV CHECKED> Watching TV<P>

<HR WIDTH=40% SIZE=6 ALIGN=left><P>

<FONT FACE=verdana COLOR=green SIZE=3><B>What are
you?</B></FONT><P>

<INPUT TYPE=radio NAME=whatru VALUE=infant> Infant<BR>
<INPUT TYPE=radio NAME=whatru VALUE=child> Child<BR>
<INPUT TYPE=radio NAME=whatru VALUE=teenager> Teenager<BR>
<INPUT TYPE=radio NAME=whatru VALUE=adult CHECKED> Adult<BR>
<INPUT TYPE=radio NAME=whatru VALUE=senior> Senior Citizen<BR>
<INPUT TYPE=radio NAME=whatru VALUE=arf> Dog<P>
```

4. Save your changes to CHECKED.HTM.
5. Toggle over to your Web browser and reload the page.

The page should appear similar to



Figure 15-2 on the following page.

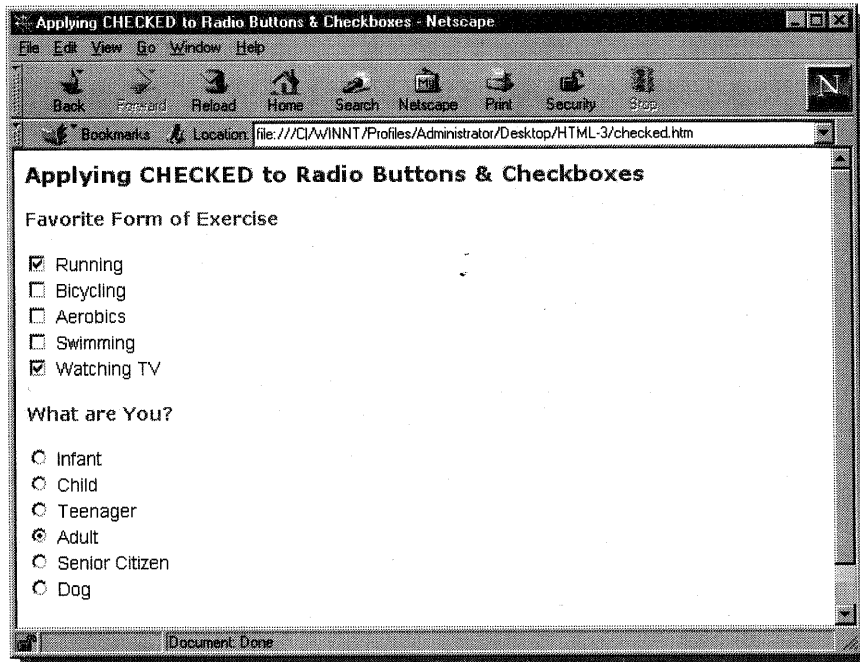


Figure 15-2: Checkboxes and radio buttons preselected using CHECKED

6. In the **Favorite Form of Exercise** section, click the checkbox adjacent to **Running**. It will become deselected.
7. Click the checkbox adjacent to **Watching TV**. It will become deselected.
8. Click the checkbox adjacent to **Swimming**. It will become selected.
9. In the **What are You** section, click **Dog**. The **Adult** radio button is deselected and the **Dog** radio button is selected.



As you can see, CHECKED only preselects options in radio button and checkbox groups. It does not in any way limit the user's choices from among the options.

List Boxes

List boxes have two supplemental attributes that can significantly change the appearance and functionality of the pulldown menu. Both supplemental attributes are described in Table 15-3.

Attribute	Value Set
SIZE	Any positive integer. Turns a "pop up" list into a scrollable list. A value of between 3 and 10 is typically best. <u>Example:</u> <code><SELECT NAME=state SIZE=6></code>
MULTIPLE	None. Simply add MULTIPLE to the opening <code><SELECT></code> tag. This will allow the user to select multiple items from the list box. MULTIPLE should be used in conjunction with SIZE. <u>Example:</u> <code><SELECT NAME=state SIZE=6 MULTIPLE></code>
SELECTED	None. Simply add SELECTED to any <code><OPTION></code> tag and that particular option will be displayed. In default list boxes that are displayed as one line that "pop up," the selected item is simply displayed in the unclicked window. When used in conjunction with SIZE, the selected option is highlighted. <u>Example:</u> <code><OPTION SELECTED>Colorado</code>

Table 15-3: List box supplemental attributes

Examples of MULTIPLE & SIZE

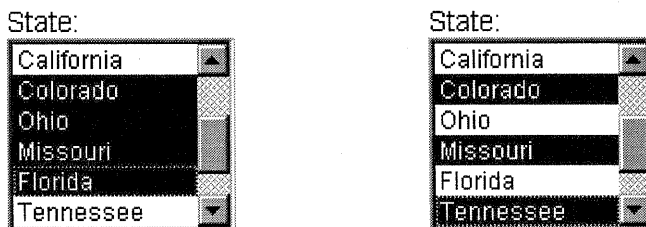


Figure 15-3: `<SELECT SIZE=6 MULTIPLE>`: contiguous block (left); discontiguous block (right)



A good example of a need to turn a pop up list into a scrollable list is a long list, such as all 50 states in the United States. Many browsers are not capable of displaying a list box of this size or they display it poorly.

Exercise 15-3: Using the SIZE and MULTIPLE Attributes

In this exercise, you will apply the `SIZE` and `MULTIPLE` attributes to a `<SELECT>` tag to create a list box that: 1) converts a pop up list to a scrollable list, displaying multiple options simultaneously, and 2) allows the user to select multiple list items in either contiguous or discontiguous groups.

1. In your Web browser, open `SIZE_AND_MULTIPLE.HTM`. The list box will appear with all defaults. (No supplemental attributes are being used with either the `<SELECT>` tag or the `<OPTION>` tag.)
2. Toggle over to Notepad and open `SIZE_AND_MULTIPLE.HTM`.
3. Add the code that appears below in bold:

What is your Internet connection bandwidth?<P>

```
<SELECT SIZE=4 NAME=bandwidth>
<OPTION>28.8 Kbps dialup modem
<OPTION>56.6 Kbps dialup modem
<OPTION>ISDN 64 Kbps
<OPTION>ISDN 128 Kbps
<OPTION>asymmetrical cable modem
<OPTION>symmetrical cable modem
<OPTION>DSL
<OPTION>satellite
<OPTION>T-1
<OPTION>T-3
</SELECT>
```

4. Save the file.
5. Toggle over to your Web browser and reload the page.



The list box should display four rows of data, as shown in Figure 15-4 on the following page.

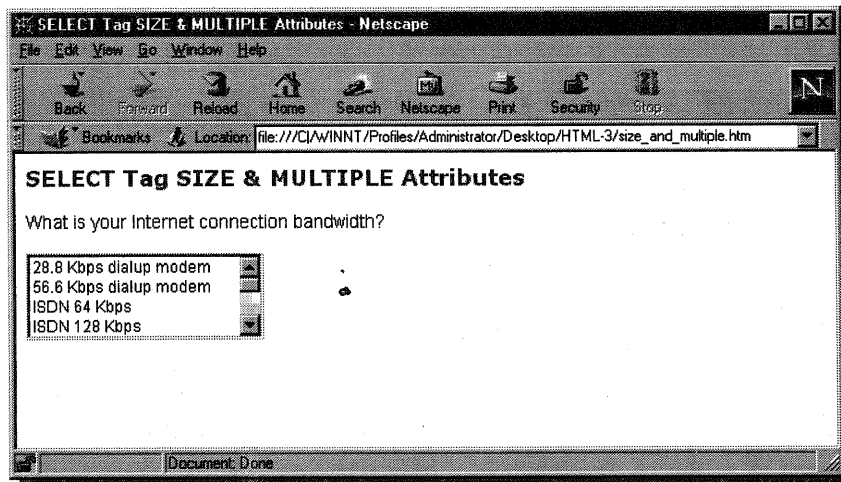


Figure 15-4: List box now displays four rows of data via the SIZE=4 attribute

6. In the list box, click the **56.6 Kbps dialup modem** list item.
7. Press and hold <CTRL> on your keyboard and click **ISDN 128 Kbps**.



The 56.6 Kbps dialup modem list item is deselected and the ISDN 128 Kbps item is selected. By default, a user cannot choose more than one item from a list box, even if the box appearance has been modified using the SIZE attribute.

8. Toggle back to Notepad.
9. Change the value of the SIZE attribute from 4 to 7. Add the following code to the opening <SELECT> tag:

MULTIPLE

10. Save the file.
11. Toggle over to your Web browser and reload the page.
12. Click the **56.6 Kbps dialup modem** list item. Hold <CTRL> on your keyboard and click **ISDN 128 Kbps**. Continue holding <CTRL> and click **DSL**. Release <CTRL>.



All three list items are selected, as shown in Figure 15-5.

When you use the **MULTIPLE** attribute, a user can select any or all items in a list box by using **<CTRL>** or **<SHIFT>**.

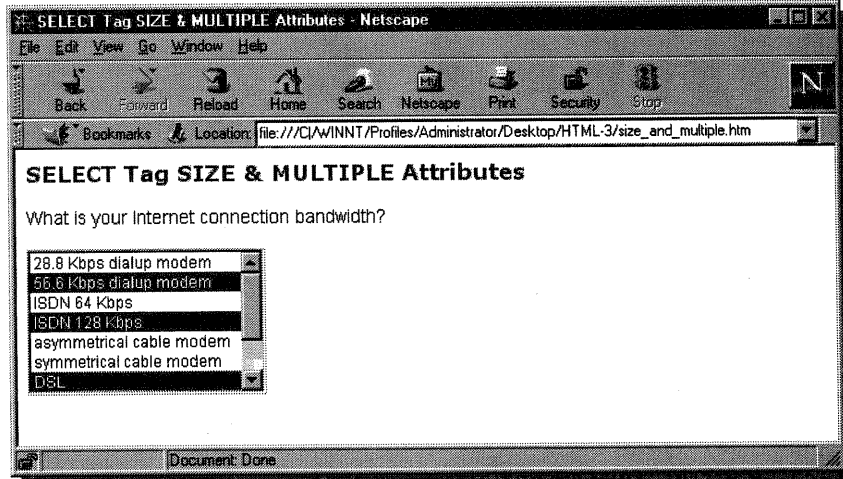


Figure 15-5: Any or all list items can be selected via the **MULTIPLE** attribute

13. In the list box, click **56.6 Kbps dialup modem**. Because you did not hold **<CTRL>** or **<SHIFT>**, the list item is selected while all other list items are deselected.
14. Press and hold **<SHIFT>** and click **symmetrical cable modem**. This will select all list items *between* the first and last items clicked, inclusive.



Many users are not aware that they have the option of choosing multiple items from a list box. Instructions describing the use of **<CTRL>** to select discontinuous items and **<SHIFT>** to select contiguous blocks may be helpful in your Web pages (especially if your target market is non-technical users).

Exercise 15-4: Using the SELECTED Attribute

In this exercise, you will apply the SELECTED attributes to a <SELECT> tag to prioritize one of the list items (options) to be displayed by default.

1. In your Web browser, open SELECTED.HTM from the HTML-3 folder on your Desktop.



A list box appears that displays seven list items, none of which are selected (this is the default for a list of any size).



Based on what you learned thus far and the characteristics of this list box, what supplemental attribute and value have been inserted into the opening <SELECT> tag?

2. In Notepad, open SELECTED.HTM.
3. Add the SELECTED attribute to the <OPTION> tag adjacent to **Football**.
4. Save the HTML document in Notepad, toggle over to your Web browser, and reload the Web page.



The list box now appears with the Football list item preselected, as shown in Figure 15-6.

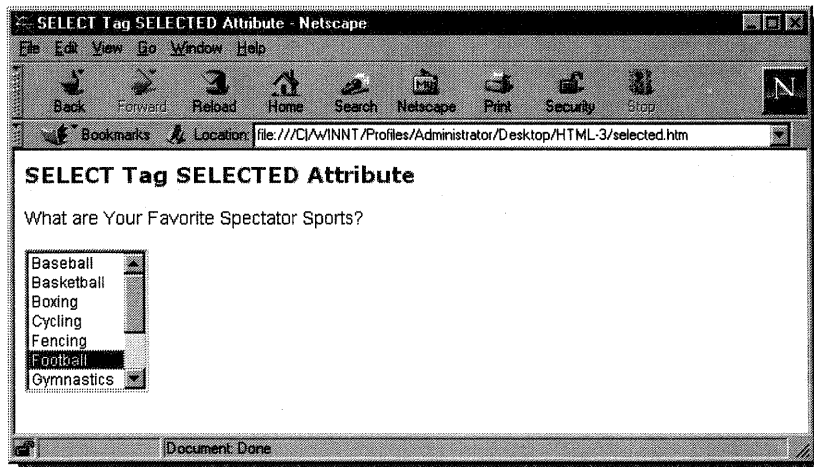


Figure 15-6: List box item preselected using the <OPTION SELECTED> attribute

5. Hold <CTRL> and click the **Baseball** list item.



The Baseball item is selected and the Football item is deselected. This is proof that the MULTIPLE attribute has not been added to the <SELECT> tag.

6. Toggle back to Notepad.
7. Add the MULTIPLE attribute to the opening <SELECT> tag. **Remember:** MULTIPLE has no value.
8. Remove the SELECTED attribute from the **Football** <OPTION> tag and add it to the **Running** list item <OPTION> tag. **Remember:** SELECTED has no value.
9. Save the HTML document, toggle over to your Web browser, and reload the page.
10. Scroll down to the bottom of the list.



The list appears with the Running list item preselected. Note that the list did *not* automatically scroll down to display the selected list item.

11. Hold <CTRL> and click **Football**. Both the preselected **Running** item and **Football** will be selected (because you added MULTIPLE in Step 7), as shown in Figure 15-7.

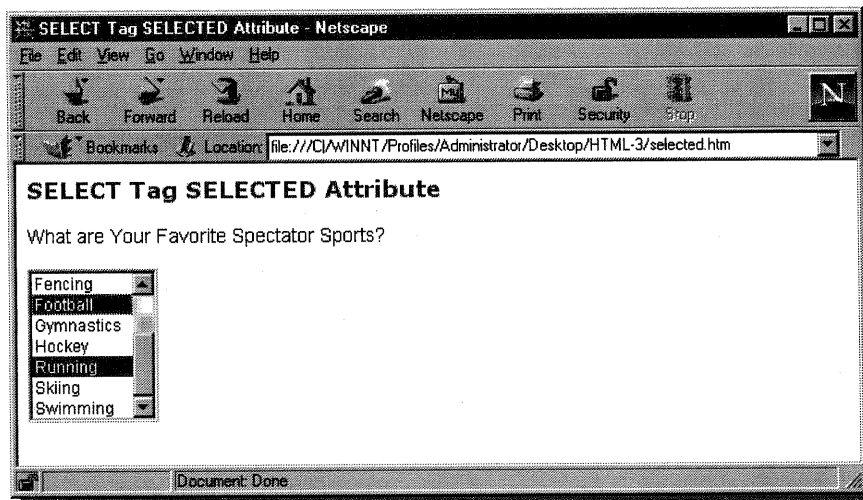


Figure 15-7: Effect of <OPTION SELECTED> attribute and <SELECT MULTIPLE>

Text Area Text

Text areas allow you to display text within the text area field. There is no special attribute or tag for this. Remember that `<TEXTAREA>` is a container.

To insert text into the text area field, you simply type it between the opening and closing `<TEXTAREA>` tags. Thus, any text that appears inside the text area container will appear inside the text area field. Note that this text can be modified by users.

Example of Modifiable Text

We welcome your comments and feedback:

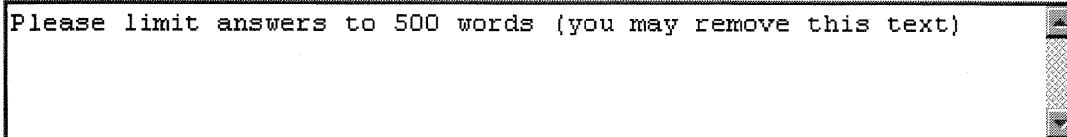


Figure 15-8: Example of modifiable text inserted into a text area field

READONLY

There is a supplemental attribute supported by the `<TEXTAREA>` tag, but it is not universally supported by all major browsers. This attribute, `READONLY`, is described in Table 15-4.

Attribute	Value Set
READONLY	<p>Makes all text in the text area read-only, so a user cannot delete or modify the text or add new text. Not allowing a user to add new text somewhat defeats the purpose of a text area.</p> <p>Supported by Internet Explorer 5.0; <i>not supported</i> by Netscape Navigator/Communicator 4.x.</p> <p><u>Example:</u></p> <pre><TEXTAREA ROWS=5 COLS=70 READONLY>This text cannot be changed</TEXTAREA></pre>

Table 15-4: `<TEXTAREA>` `READONLY` attribute



`READONLY` can be applied to any `<INPUT>` tag. Thus, one application of `READONLY` could be the creation of online “sample” forms in which you do not want to receive input from a user.

Non-Web examples are published by insurance companies and hospitals, with large watermarks across them to denote they are samples.

Exercise 15-5: Adding Modifiable Text to a Text Area Field

In this exercise, you will add modifiable text to a text area field that provides additional instructions or direction to any user of your form.

1. In your Web browser, open TEXTAREA.HTM. Note the appearance of the text area field.
2. In Notepad, open TEXTAREA.HTM.
3. Add the following code that appears in bold:

```
<TEXTAREA NAME=comments ROWS=5 COLS=65>Please limit answers to  
500 words (you may remove this text)</TEXTAREA>
```

4. Save the HTML document.
5. Toggle over to your Web browser and reload the Web page.



The text area field will now display the text you typed between the opening and closing `<TEXTAREA>` tags in Step 3 (known as the text area container), as shown in Figure 15-9.

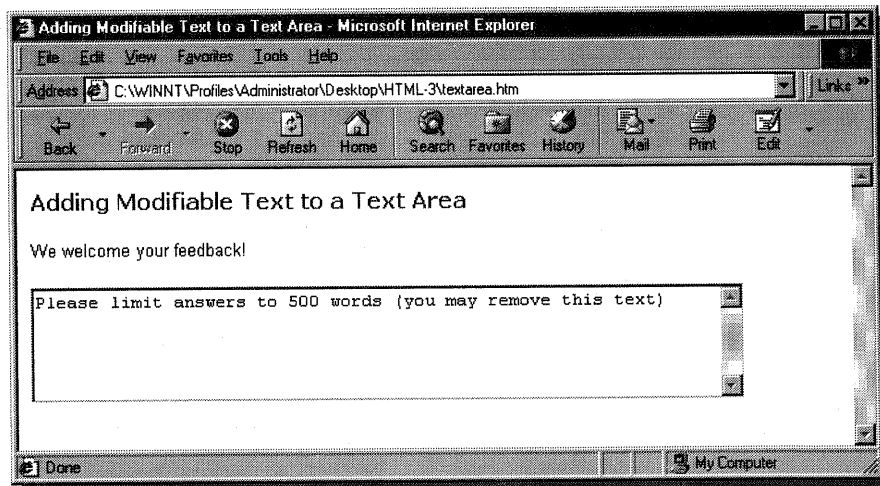


Figure 15-9: Text added inside text area field

Exercise 15-6: Adding READONLY Text to a Text Area Field

In this exercise, you will add READONLY text to a text area field. This text cannot be modified by a user, nor can additional text be added to the text area field. For this reason, the READONLY attribute is rarely added to the <TEXTAREA> tag.

1. In your Web browser, open TEXTAREA-2.HTM. Note the appearance of the text area.
2. In Notepad, open TEXTAREA-2.HTM.
3. Add the following code to the opening <TEXTAREA> tag: **READONLY**.
4. Change the text between the opening and closing <TEXTAREA> tags to read “**--This text cannot be altered--**”.
5. Save the HTML document.
6. Toggle over to your Web browser and reload the Web page.
7. Attempt to select the text in the text area field. Attempt to type additional text in the field.



The text inside the text area field cannot be altered and no additional text can be added, as shown in Figure 15-10.

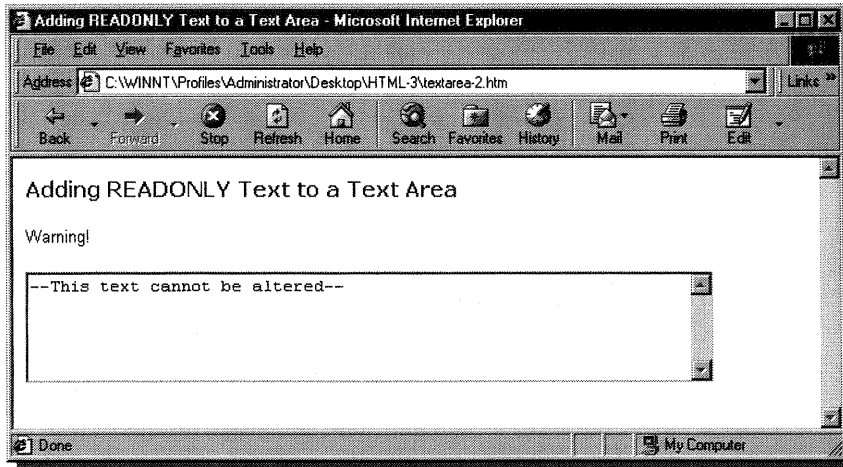


Figure 15-10: Text area field with unmodifiable text due to READONLY attribute



READONLY is not properly interpreted by Netscape Navigator/Communicator. Therefore, this attribute works only with Microsoft Internet Explorer or any fully compliant HTML 4.0 browser.

Submit & Reset Buttons

You may have already wished you could change the text that is displayed on the Submit and Reset buttons. "Submit Query" is not logical for many applications. You may also have noticed that other Web sites display the Submit and Reset buttons with customized text.

You can, in fact, modify the text displayed on the face of the Submit and Reset buttons to the text of your choice. As you will see later in this Lesson, you can also substitute graphics for the buttons while retaining full functionality.

Example



Figure 15-11: Customized Submit and Reset button faces

The supplemental attribute that creates this effect is `VALUE`, as described in Table 15-5.

Attribute	Value Set
VALUE	<p>Changes the text that is displayed on the face of the Submit or Reset buttons. The value of <code>VALUE</code> is case sensitive.</p> <p><u>Example:</u></p> <pre><INPUT TYPE=submit VALUE=Upload Survey></pre>

Table 15-5: `VALUE` attribute to the Submit and Reset `<INPUT>` tags



DDC's *Mastering HTML 4.0* Series does not follow the official HTML 4.0 Specification with regard to attribute values, which the official Specification states should always be enclosed in quotation marks (exam: `TYPE="radio"`).

Going back several versions, both Netscape Navigator/Communicator and Microsoft Internet Explorer properly interpret attribute values without the quotation marks.

There is one important exception, however: when the attribute value is composed of more than one word. In such cases, you must always use quotations marks around the value or only the first word will be interpreted (e.g.: ``).

Exercise 15-7: Customizing Submit and Reset Button Faces

In this exercise, you will customize the text displayed on the faces of the Submit and Reset buttons by using the `VALUE` attribute to the Submit and Reset `<INPUT>` tags.

1. In Notepad, open `BUTTON_VALUE.HTM` from the `HTML-3` folder on your Desktop.
2. Scroll to the bottom of the HTML document and note the standard code for the Submit and Reset buttons.
3. Toggle over to your Web browser and open `BUTTON_VALUE.HTM`. Scroll to the bottom of the form and note the standard appearance of the Submit and Reset buttons.
4. Toggle over to Notepad and add the following code at the bottom of the document:

```
<INPUT TYPE=submit VALUE="Upload Application"> <INPUT TYPE=reset  
VALUE="Clear Form">
```

5. Save your changes to the HTML document.
6. Toggle over to your Web browser and reload the Web page.



The Submit and Reset buttons will now appear with the new text you specified by adding the `VALUE` attributes to each tag, as shown in Figure 15-12.

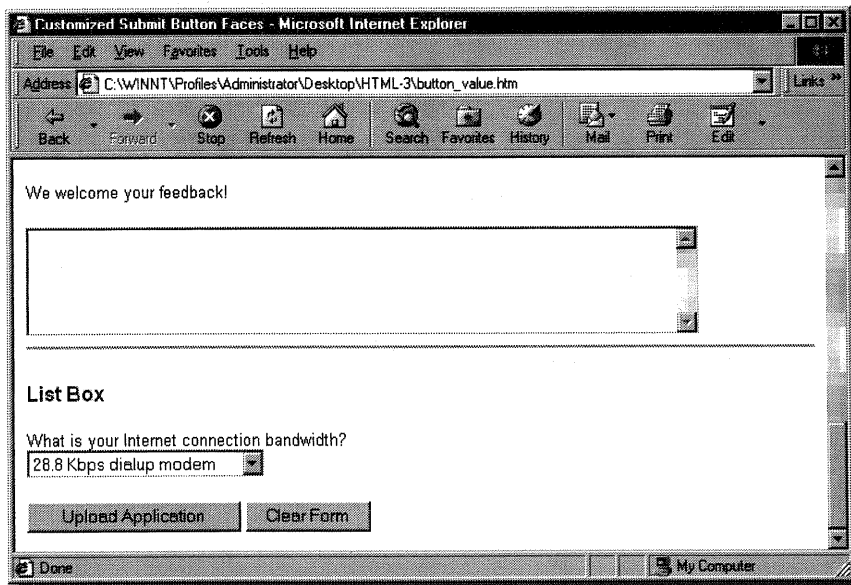


Figure 15-12: Submit and Reset buttons customized using `VALUE` attribute

Using an Image for the Submit Button

You already know that you can embed any HTML element into a form, including images. However, you can also use an image to replace the Submit button while retaining its functionality.



You cannot use an image in place of the Reset button. Typically, if an image is used for the Submit function, no Reset button is provided.

Many forms, regardless of whether they use an image for the Submit button, do not offer a Reset button. Note that there is no technical requirement for a Reset button; you never have to include a Reset button if you do not want one.

Process

To use an image as a Submit button, do the following:

- specify the value of the `TYPE` attribute as `image` (`TYPE=image`)
- add a `SRC` attribute, the value of which is the name of the image file referenced
- add an optional `BORDER` attribute and specify a value of zero (`BORDER=0`); this will prevent a blue border from appearing around the image.

Code

```
<INPUT TYPE=image SRC=upload_survey.gif BORDER=0>
```



You can specify a GIF, JPEG, or PNG image as the value of the `SRC` attribute, which is identical to the use of the `` tag.

Exercise 15-8: Substituting an Image for the Submit Button

In this exercise, you will substitute a GIF image for the Submit button. Note that you cannot substitute an image for the Reset button.

1. In Notepad, open SUBMIT_IMAGE.HTM from the HTML-3 folder on your Desktop.
2. Remove the following HTML code from the bottom of the document:

```
<INPUT TYPE=submit> <INPUT TYPE=reset>.
```

3. Add the following code in place of the code you removed in Step 2:

```
<INPUT TYPE=image SRC=upload_survey.gif BORDER=0>
```

4. Save the HTML document.
5. Toggle over to your Web browser and open SUBMIT_IMAGE.HTM.



The page appears with no Reset button and an image in place of the Submit button, as shown in Figure 15-13.

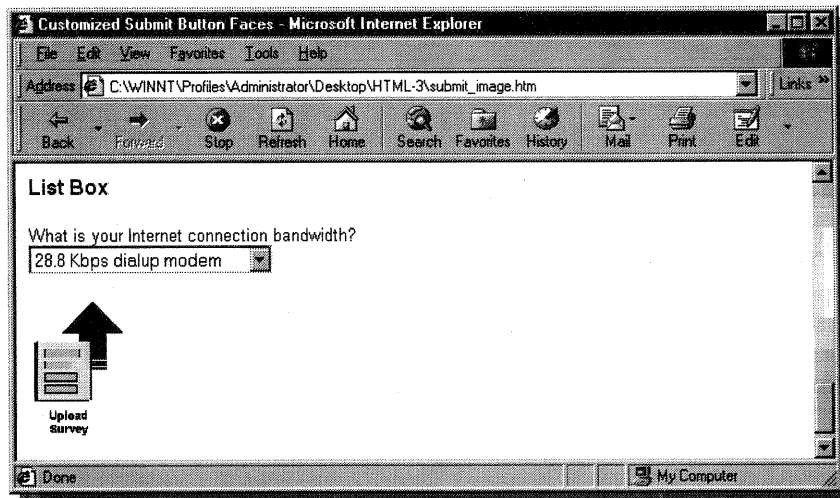


Figure 15-13: Submit button replaced with a GIF image

6. Click the **Upload Survey** image. The form will be uploaded according to the attributes in the opening `<FORM>` tag. Is this form going to an e-mail account or a CGI script?

Embedding Tables in Forms

You can add significant structure and order to a form by embedding tables. Figure 15-14 features an examples of the same form, with and without an embedded table.

Salutation: <input type="text" value="Mr."/>	Salutation: <input type="text" value="Mr."/>
First Name: <input type="text" value="Curt"/>	First Name: <input type="text" value="Curt"/>
Middle Initial: <input type="text" value="A"/>	Middle Initial: <input type="text" value="A"/>
Last Name: <input type="text" value="Robbins"/>	Last Name: <input type="text" value="Robbins"/>
Address: <input type="text" value="420 West Main St."/>	Address: <input type="text" value="420 West Main St."/>
City: <input type="text" value="Purcellville"/>	City: <input type="text" value="Purcellville"/>
State: <input type="text" value="Virginia"/>	State: <input type="text" value="Virginia"/>
ZIP Code: <input type="text" value="20132-3166"/>	ZIP Code: <input type="text" value="20132-3166"/>

Figure 15-14: Form without a table embedded (left) and with (right)

Code

```
<FORM ACTION=mailto:webmaster@corp.com METHOD=post>
<TABLE>
<TR><TD>Salutation:</TD> <TD><SELECT NAME=title>
    <OPTION>Mr.
    <OPTION>Mrs.
    <OPTION>Ms.
    <OPTION>Dr.
  </SELECT></TD></TR>
<TR><TD>First Name:</TD> <TD><INPUT TYPE=text NAME=first_name
SIZE=12></TD></TR>
<TR><TD>Middle Initial:</TD> <TD><INPUT TYPE=text SIZE=2
NAME=mid_initial></TD></TR>
<TR><TD>Last Name:</TD> <TD><INPUT TYPE=text NAME=last_name
SIZE=15></TD></TR>
<TR><TD>Address:</TD> <TD><INPUT TYPE=text NAME=address></TD></TR>
<TR><TD>City:</TD> <TD><INPUT TYPE=text NAME=city></TD></TR>
<TR><TD>State:</TD> <TD><SELECT NAME=state>
    <OPTION>Alabama
    <OPTION>Ohio
    <OPTION>Colorado
    <OPTION>Virginia
  </SELECT></TD></TR>
<TR><TD>ZIP Code:</TD> <TD><INPUT TYPE=text SIZE=10
NAME=zip></TD></TR>
</TABLE><P>
```

One Step Further

You could take the above example one step further and right align the text labels for the form fields to give it a nice Desktop publishing effect. This is easy to accomplish; simply add `ALIGN=right` attributes to the opening `<TD>` tag of each text label.

An example of this is shown below; the code behind it follows.

Salutation:	<input type="text" value="Mr."/>	Salutation:	<input type="text" value="Mr."/>
First Name:	<input type="text" value="Curt"/>	First Name:	<input type="text" value="Curt"/>
Middle Initial:	<input type="text" value="A."/>	Middle Initial:	<input type="text" value="A."/>
Last Name:	<input type="text" value="Robbins"/>	Last Name:	<input type="text" value="Robbins"/>
Address:	<input type="text" value="420 West Main St."/>	Address:	<input type="text" value="420 West Main St."/>
City:	<input type="text" value="Purcellville"/>	City:	<input type="text" value="Purcellville"/>
State:	<input type="text" value="Virginia"/>	State:	<input type="text" value="Virginia"/>
ZIP Code:	<input type="text" value="20132-3166"/>	ZIP Code:	<input type="text" value="20132-3166"/>

Figure 15-15: Table cells default left alignment (left) and enhanced with right alignment (right)

```
<TABLE>
<TR><TD ALIGN=right>Salutation:</TD> <TD><SELECT NAME=title>
  <OPTION>Mr.
  <OPTION>Mrs.
  <OPTION>Ms.
  <OPTION>Dr.
</SELECT></TD></TR>
<TR><TD ALIGN=right>First Name:</TD> <TD><INPUT TYPE=text
NAME=first_name SIZE=12></TD></TR>
<TR><TD ALIGN=right>Middle Initial:</TD> <TD><INPUT TYPE=text
SIZE=2 NAME=mid_initial></TD></TR>
<TR><TD ALIGN=right>Last Name:</TD> <TD><INPUT TYPE=text
NAME=last_name SIZE=15></TD></TR>
<TR><TD ALIGN=right>Address:</TD> <TD><INPUT TYPE=text
NAME=address></TD></TR>
<TR><TD ALIGN=right>City:</TD> <TD><INPUT TYPE=text
NAME=city></TD></TR>
<TR><TD ALIGN=right>State:</TD> <TD><SELECT NAME=state>
  <OPTION>Alabama
  <OPTION>Ohio
  <OPTION>Colorado
  <OPTION>Virginia
</SELECT></TD></TR>
<TR><TD ALIGN=right>ZIP Code:</TD> <TD><INPUT TYPE=text SIZE=10
NAME=zip></TD></TR>
</TABLE><P>
```

Exercise 15-9: Embedding a Table in a Form to Add Structure

In this exercise, you will add structure to a form by embedding a table. You will also enhance the table by right aligning the text labels in the left column.

1. In Notepad, open ADD_TABLE.HTM from the HTML-3 folder on your Desktop.
2. Toggle over to your browser and open ADD_TABLE.HTM. Note the appearance of the form fields in relation to the field labels.
3. Toggle back to Notepad and add the following code that appears in bold:

```
<TABLE>
```

```
<TR><TD ALIGN=right>Salutation:</TD> <TD><SELECT NAME=title>  
    <OPTION>Mr.  
    <OPTION>Mrs.  
    <OPTION>Ms.  
    <OPTION>Dr.  
</SELECT></TD></TR>
```

```
<TR><TD ALIGN=right>First Name:</TD> <TD><INPUT TYPE=text  
NAME=first_name SIZE=12></TD></TR>
```

```
<TR><TD ALIGN=right>Middle Initial:</TD> <TD><INPUT TYPE=text  
SIZE=2 NAME=middle_initial></TD></TR>
```

```
<TR><TD ALIGN=right>Last Name:</TD> <TD><INPUT TYPE=text  
NAME=last_name SIZE=15></TD></TR>
```

```
<TR><TD ALIGN=right>Address:</TD> <TD><INPUT TYPE=text  
NAME=address></TD></TR>
```

```
<TR><TD ALIGN=right>City:</TD> <TD><INPUT TYPE=text  
NAME=city></TD></TR>
```

```
<TR><TD ALIGN=right>State:</TD> <TD><SELECT NAME=state>  
    <OPTION>Alabama  
    <OPTION>Ohio  
    <OPTION>Colorado  
    <OPTION>Virginia  
</SELECT></TD></TR>
```

```
<TR><TD ALIGN=right>ZIP Code:</TD> <TD><INPUT TYPE=text SIZE=10  
NAME=zip></TD></TR>
```

```
</TABLE><P>
```

4. Save ADD_TABLE.HTM.
5. Toggle over to your Web browser and reload the page.



The form with the embedded table appears in your browser, as shown in Figure 15-16.

Figure 15-16: Form with an embedded table to give structure to fields



For a review of tables, see DDC's *HTML 4.0 Intermediate*.



You can also embed forms within forms. Although there is rarely a need to do this, it is important to know the capabilities of forms so you can meet any scripting or Web publishing challenge.

Enhancing Tables Embedded in Forms

You can change the color of body text in any HTML element, including forms. But to give some real graphic flair to a form, you can embed a table and color its cells.

In many respects, forms and tables are made to be co-mingled. The form shown in Figure 15-17 is the result of many different modifications to a standard form, including:

- an embedded table
 - `<TD ALIGN=right>` in the first column of rows 2 through 9
 - table cells filled with color with `BGCOLOR=blue` attribute to the `<TD>` tags
 - `CELLPADDING=6` and `CELLSPACING=6`
 - `WIDTH=75%`
 - `COLSPAN=2` for the first and last rows
- `` applied to each field label in the left column
- form and table centered on the page with `<CENTER>` tag set
- Submit button face modified (using the `VALUE` attribute) and aligned right

The screenshot shows a Netscape browser window with the title 'Adding a Table to a Form - Netscape'. The address bar shows the file path: file:///C:/WINNT/Profiles/Administrator/Desktop/HTML-3/add_table.htm. The main content area displays a form titled 'Open a New Account with Us'. The form is structured as a table with two columns. The first column contains labels for various fields, and the second column contains the input fields. The labels are: Salutation, First Name, Middle Initial, Last Name, Address, City, State, and ZIP Code. The input fields are: a dropdown menu for Salutation (set to Mr.), text boxes for First Name (Curt), Middle Initial (A), Last Name (Robbins), Address (410 West Main St.), City (Purcellville), State (Virginia), and ZIP Code (20132-3166). A 'Submit the Application' button is located at the bottom right of the form. The table has a blue background and is centered on the page.

Figure 15-17: Enhanced table embedded in a form

Exercise 15-10: Enhancing a Table Embedded in a Form

In this exercise, you will enhance a table embedded in a form to give it more visual appeal. Much of this exercise should be a review of what you learned about tables in DDC's *HTML 4.0 Intermediate*.

1. In your Web browser, open TABLE-2.HTM from the HTML-3 folder on your Desktop. Note the appearance of the form fields.
2. Toggle over to Notepad and open TABLE-2.HTM.
3. Modify the following code that appears in bold:

```
<TABLE WIDTH=75% CELSPACING=6 CELLPADDING=6>
<TR><TD BGCOLOR=gray COLSPAN=2><FONT SIZE=4 FACE=verdana
COLOR=white><B>Open a New Account with Us</B></FONT>
</TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>
Salutation:</FONT></B></TD> <TD BGCOLOR=green>

<SELECT NAME=title>
  <OPTION>Mr.
  <OPTION>Mrs.
  <OPTION>Ms.
  <OPTION>Dr.
</SELECT></TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>First
Name:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text
NAME=first_name SIZE=12></TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>Middle
Initial:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text
SIZE=2 NAME=middle_initial></TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>Last
Name:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text
NAME=last_name SIZE=15></TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>
Address:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text
NAME=address>
</TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>
City:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text
NAME=city></TD></TR>
<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>
State:</FONT></B></TD> <TD BGCOLOR=green>
```

```

<SELECT NAME=state>
  <OPTION>Alabama
  <OPTION>Alaska
  <OPTION>Arizona
  others intentionally not listed here...
</SELECT></TD></TR>

<TR><TD BGCOLOR=blue ALIGN=right><B><FONT COLOR=white>ZIP
Code:</FONT></B></TD> <TD BGCOLOR=green><INPUT TYPE=text SIZE=10
NAME=zip></TD></TR>

<TR><TD COLSPAN=2 ALIGN=right><INPUT VALUE="Submit the
Application" TYPE=submit></TD>

```

- Note that the ACTION attribute in the opening <FORM> tag has no value. Specify a value that will send the output of the form to your e-mail account (replacing the “???”).
- Save the HTML file, toggle over to your Web browser, and reload the page.



The form will appear with the table enhancements, significantly increasing its graphic appeal and professionalism, as shown in Figure 15-18.

Enhancing a Table Embedded in a Form - Microsoft Internet Explorer

Address C:\WINNT\Profiles\Administrator\Desktop\HTML 3\table-2.htm

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit

Open a New Account with Us

Salutation:	Mr.
First Name:	Curt
Middle Initial:	A
Last Name:	Robbins
Address:	420 West Main St
City:	Purcellville
State:	Virginia
ZIP Code:	20132-3166

Submit the Application

Done My Computer

Figure 15-18: Enhancing a form with a formatted table

Forms Tips & Tricks

This section provides you with several tips and tricks that you might not otherwise know or learn for several months or even years of scripting HTML forms. As with all elements of HTML, there are special techniques that can save you significant time or give you a competitive edge over competing Web sites.

Posting a Form to Multiple E-mail Recipients

It is easy to send the output of a form to multiple e-mail accounts. You can even determine if form output is sent as a direct e-mail message or as a carbon copy.

Code

```
<FORM ACTION=mailto:person1@corp.com?cc=person2@corp.com
METHOD=post>
```

Note the question mark, “?”, that separates the primary (first) e-mail recipient from the second, carbon copy recipient. Also note the “cc” denotation to signify that person2@corp.com is receiving a carbon copy message of the form content.

Three or More Recipients

It is possible to send a carbon copy and blind carbon copy to more than one e-mail account, allowing a total of three or more recipients of the posted form. Simply add the additional e-mail addresses, but do not use the “?” symbol. Instead, use the ampersand “&,” as shown below:

```
<FORM ACTION=mailto:person2@corp.com?cc=person2@corp.com&cc=
person2@corp.com&cc=person2@corp.com&bcc=person4@corp.com
METHOD=post>
```



Receipt of forms by multiple e-mail addresses works only in IE 4.0 and above and Netscape Navigator/Communicator 4.0 and above.

In fact, the MAILTO: protocol itself works only in version 4.0 and above of the major browsers.

Readable MAILTO Output

Many Webmasters and HTML scripters learn the hard way what is otherwise an easy trick. This trick makes the output of a form posted to an e-mail account much easier to read. It is a small `<FORM>` tag attribute called `ENCTYPE`, which stands for *Encoding Type*.

Code

```
<FORM ACTION=mailto:person1@corp.com METHOD=post ENCTYPE=text/plain>
```

The `ENCTYPE` attribute tells the server to separate the different form fields, making them substantially easier to read.

Figure 15-19 displays an example of form output submitted to an e-mail account in which the opening `<FORM>` tag did *not* have the `ENCTYPE` attribute.

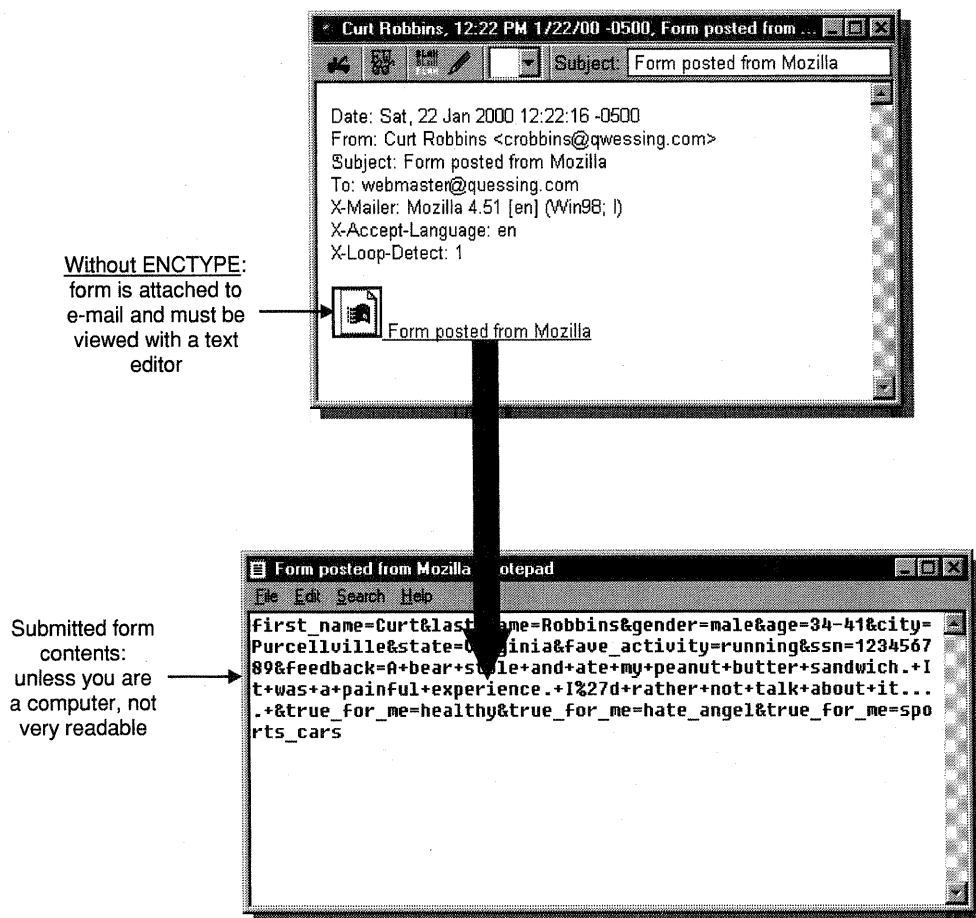


Figure 15-19: Form with `<FORM>` tag lacking `ENCTYPE` attribute

Figure 15-20 displays the exact same form with identical data populating the fields. This output, however, was submitted to an e-mail account using the ENCTYPE attribute.

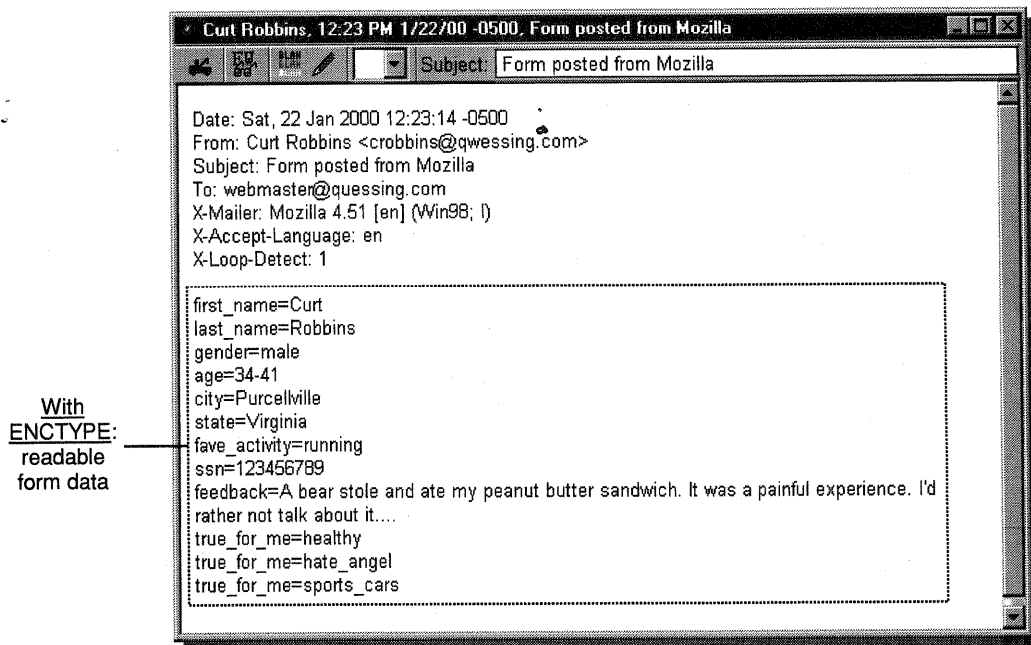


Figure 15-20: Form output with <FORM> tag and ENCTYPE=text/plain attribute

Right Align Text in any <INPUT> Field

You will learn more about *Cascading Style Sheets*, better known as CSS, later in this course. For now, you only need to know that a small element of CSS can be applied to forms to right align the contents of any <INPUT> field.

Code

```
<INPUT TYPE=text NAME=address STYLE=text-align:right>
```

An example of a form with the contents of all <INPUT> fields right aligned using the CSS STYLE attribute is shown in Figure 15-21.

Using CSS to Right Align <INPUT> Field Contents - Microsoft Internet Explorer

Address D:\Courseware\HTML 4.0 Advanced\css right align.htm

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit

Right Aligning Input Field Contents

First Name:

Last Name:

Address:

City:

Province:

Postal Code:

Credit Card Number:

Social Security Number:

Done My Computer

Right-aligned
field content
using CSS
STYLE attribute

Figure 15-21: Right aligning form field contents using the CSS STYLE attribute

Posting Forms to CGI Programs

It should be noted that CGI is a complex topic and could occupy several full days of training. The purpose of this section is simply to inform you of the potential applications and basic characteristics of CGI programs that can accept output from your forms.

Note: the goal of this section is not to teach you how to write CGI programs. For an in-depth review of CGI, see DDC's two-day *CGI/Perl Fundamentals* course.

CGI Functions

As you already know, a CGI program can be written to perform almost any function or action when it receives the posted output from a form. The most common CGI program roles are:

- Database interface: a CGI program can take form output, format it, and submit it to a database for archival or action purposes. A single CGI program can submit data from a single form in various formats to multiple databases.
- User feedback: a CGI program can take form output and respond to a user in some manner by downloading an HTML document to the browser, as shown in Figure 15-22. Typically, the type and nature of the HTML document downloaded are specific to the data populating the form fields submitted.
- Perform calculations: a CGI program can take data submitted with the form and calculate it in some manner. The CGI program may or may not then download a Web page reporting the results of this calculation.

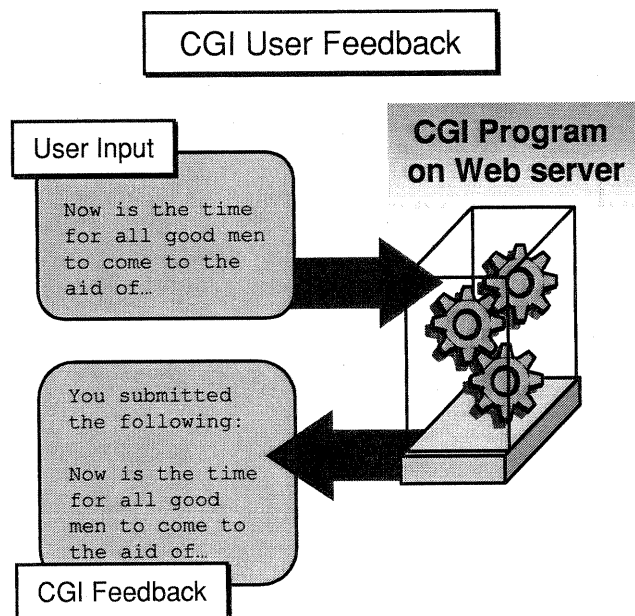


Figure 15-22: Example of CGI program that provides user feedback

Example: An Online Test

Sometimes, calculations are performed before the results are posted to a database (regardless of whether feedback is provided to the user). For example, assume a user takes a test online and it is administered as an HTML form.

The process is as follows:

1. User populates form fields by answering the questions.
2. User uploads the completed “test” (form).
3. The CGI program calculates which answers the user got correct and which she missed.
4. The CGI program then summarizes the details according to the desires of the programmers:
 - CGI program may want to document exactly which questions were missed.
 - CGI program may download to the user more questions which are pulled from a database and based on the ones the user missed.
 - CGI program may only calculate the user’s percentage of correct answers with no follow up action.
5. After the results are summarized, the data is transferred to a “backend” database.
6. Other applications—separate from the CGI program—may perform operations with the data after it is archived in the database, such as a reporting function or follow up action.

Example: User Feedback

Figure 15-23 shows an example of a simple CGI script residing on a Web server that takes data posted from a form and returns it to the user who submitted the form as a Web page.

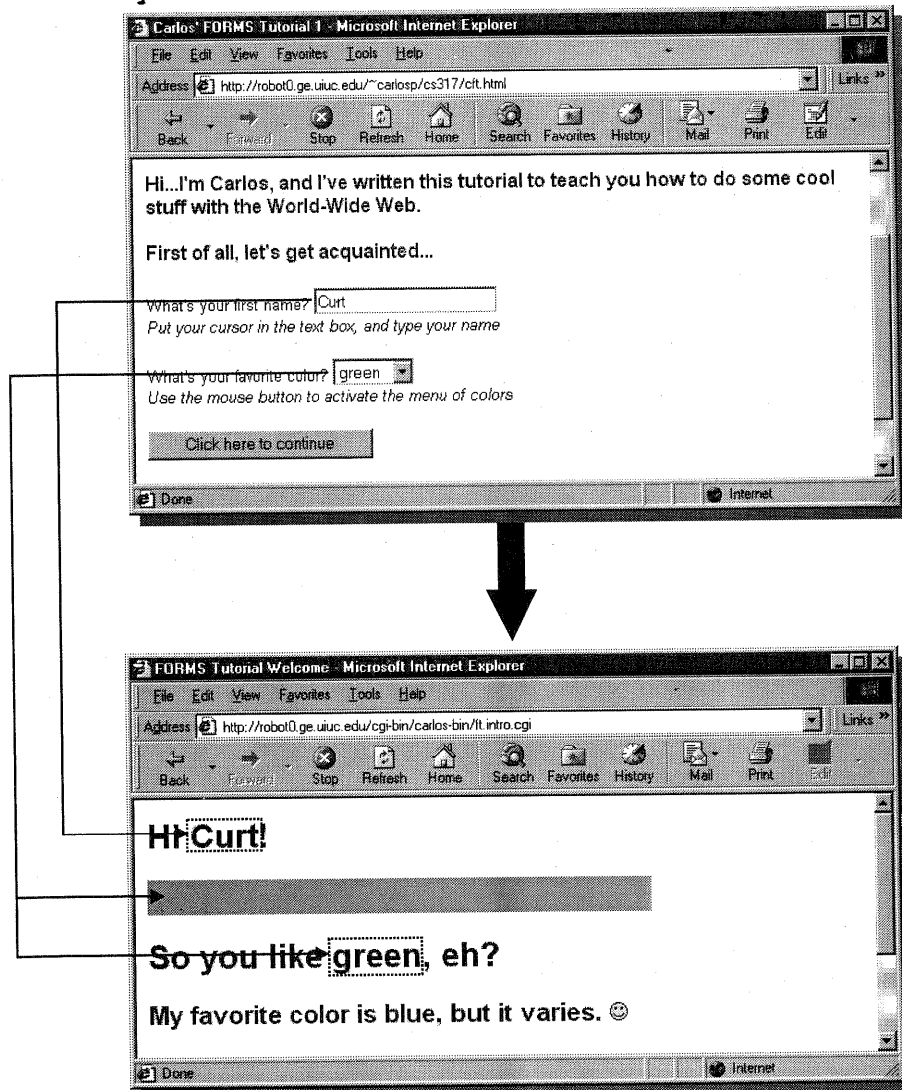


Figure 15-23: Example of a simple CGI program that returns a Web page to a user

Exercise 15-11: Using Forms that Access CGI Scripts

In this exercise, you will populate and upload multiple forms that access CGI echo scripts. This certainly does not give you the ability to write your own CGI scripts, but it does give you a better idea of the fact that forms are only part of a bigger overall system involving CGI scripts and other server-level processing mechanisms.

1. Access the following URL: `robot0.ge.uiuc.edu/~carlosp/cs317/cft.html`.
2. In the **What's your first name?** field, type your name.
3. In the **What's your favorite color?** list box, select the color of your choice.
4. Click the **Click here to continue** button. This is the **Submit** button for the form. (Remember: you can use the `VALUE` attribute to change the face of the Submit button to display anything you want).



A CGI script residing on a Web server uses the data you uploaded to populate particular fields in a customized Web page that it automatically downloads to your browser.

5. Access the following URL: `members.whro.org/~pterry/cgi-bin/form2.shtml`.
6. Input your information in the **First Name**, **Last Name**, and **Email Add** fields.
7. Click the **Send Form** button.



A different CGI script uses your submitted data to create a custom Web page and automatically download it to you. Note that the page also has identified the state from which you are connected to the Internet (this does not always function properly).

8. Access `www.video.ufl.edu/~yakcheez/samples/psample6.html`.
9. In the **Red**, **Green**, and **Blue** fields, type **22**, **33**, and **44**, respectively. Press **Compute**.



A CGI script calculates the hexadecimal value of the combined numbers that you uploaded. Unlike the previous examples, this is a good example of a calculation performed by a CGI script.

Exercise 15-12: Posting Form Output to a CGI Script

In this exercise, you will change the `ACTION` value of the opening `<FORM>` tag to post data to a CGI script. The script, a simple echo parser, will automatically download a Web page to the user based on the content of the populated form fields.

1. In Notepad, open TEST-CGI.HTM from the HTML-3 folder on your Desktop.
2. Add the following code that appears in bold:

```
<H2>Uploading to a CGI Script</H2>
```

```
<FORM ACTION= http://sj.znet.com/~aleatory/quessing/post-query.pl  
METHOD=post>
```

```
<H3>Basic Text Field</H3>
```

3. Save the file.
4. Toggle over to your Web browser and open TEST-CGI.HTM.
5. Populate the fields with your personal data. When you are done, click the **Upload to CGI Script** submit button at the bottom of the form.



The form is submitted to a CGI script residing on a Web server at the University of Illinois. The CGI script will process the form, taking the populated content of the form fields, reformatting it in an HTML document, and downloading it to you, as shown in Figure 15-24.

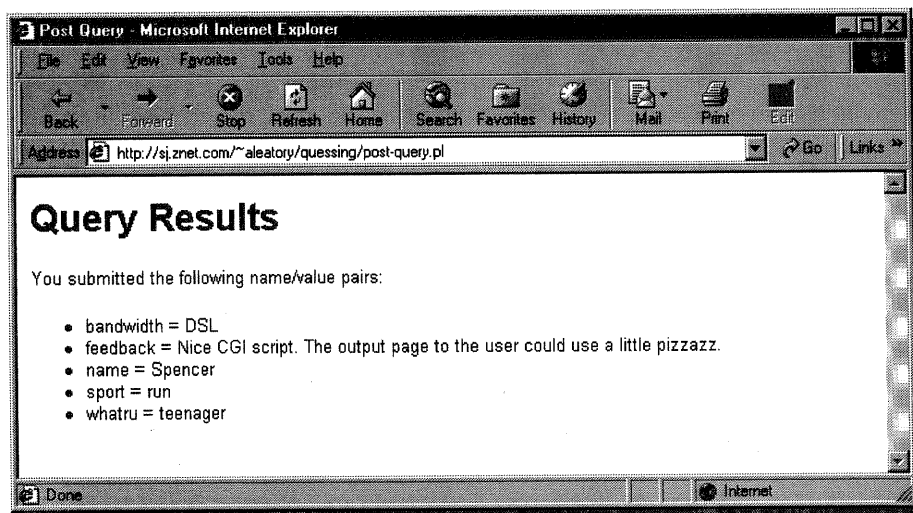


Figure 15-24: Example of output from a server-based CGI script

Lesson 15 Summary

- ▶ There are many supplemental attributes available to enhance and expand the functionality of your forms.
- ▶ The `<INPUT>` tag, when used with text and password form fields, offers `SIZE` and `MAXLENGTH` as supplemental attributes. `SIZE` can be any positive integer and translates into the width of the text field in characters. `MAXLENGTH` is also any positive integer and limits the number of characters that a user can input into the field. Technically, `SIZE` and `MAXLENGTH` do not affect one another.
- ▶ The `<INPUT>` tag, when used with checkboxes and radio buttons, has a supplemental attribute of `CHECKED`, which has no value set. `CHECKED` makes a checkbox or radio button preselected with no user action. The user can still deselect or reselect the field.
- ▶ The `<SELECT>` tag, used to create list boxes, has three supplemental attributes: `SIZE`, `MULTIPLE`, and `SELECTED`. `SIZE` can be any positive integer and turns a “pop up” list into a scrollable list. A `SIZE` value of between 3 and 10 is typically best. `MULTIPLE` has no value set (similar to `CHECKED` for radio buttons and checkboxes). It allows the user to select multiple items from the list box (using `<SHIFT>` or `<CTRL>`).
- ▶ `SELECTED` also has no value set. Unlike `SIZE` and `MULTIPLE`, `SELECTED` is applied to one or more `<OPTION>` tags within a `<SELECT>` container. If the list box is configured without the `SIZE` attribute and is a single line, the `<OPTION>` item tagged with the `SELECTED` attribute will be the only one displayed in the list box. If the list box is configured as a scroll box, any list items tagged with `SELECTED` will be highlighted.
- ▶ A rarely used supplemental attribute to the `<TEXTAREA>` tag is `READONLY`. `READONLY` makes all text in the text area read only, so a user cannot delete or modify the text or add new text.
- ▶ You can change the text that appears on the face of the Submit and Reset buttons using the `VALUE` attribute. When added to the `<INPUT>` tags, the value of `VALUE` will be displayed. Note: the value of `VALUE` is case sensitive.
- ▶ You can use an image in place of the Submit button. You cannot use an image in place of the Reset button. There is no technical requirement for a Reset button. To use an image in place of the Submit button, use the following HTML syntax: `<INPUT TYPE=image SRC=filename.gif BORDER=0>`.
- ▶ One way in which you can significantly enhance the visual appeal and user-friendliness of your forms is to embed HTML tables in them. Remember: you can embed any HTML element inside a `<FORM>` container, including frames, forms, and tables.
- ▶ You can post a form to multiple e-mail accounts using the `MAILTO` protocol to the `ACTION` attribute of the opening `<FORM>` tag.
- ▶ You should use `ENCTYPE=text/plain` in the `<FORM>` tags of your `MAILTO` forms.